(19) Europäisches Pat ntamt
Europ an Pat nt Offi
Office europé n d s br v ts

(11) EP 0 887 981 A2

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
30.12.1998 Bulletin 1998/53

(51) Int Cl.6: H04L 29/06

(21) Application number: 98304869.5

(22) Date of filing: 19.06.1998

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: 26.06.1997 US 883636

(71) Applicant: SUN MICROSYSTEMS, INC.
Mountain View, California 94043-1100 (US)

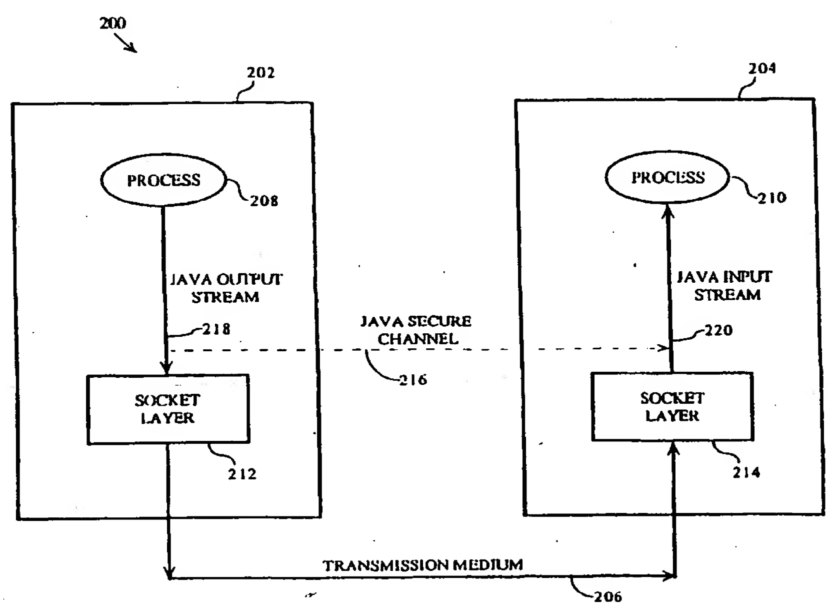
(72) Inventor: Gong, Li
Menlo Park, California 94025 (US)

(74) Representative: Read, Matthew Charles et al
Venner Shipley & Co.
20 Little Britain
London EC1A 7DH (GB)

# (54) Layer-independent security for communication channels

(57) A method and apparatus for providing layer-independent secure network communication is provided. According to an embodiment of the invention, a transmission medium is provided between a first network node and a second network node. Both the first network node and the second network node support at least one common communication protocol. A Java output stream is established between a first process executing on the first network node and the transmission medium. Also, a Java input stream is established between a second process executing on the second multilayered node and the transmission medium. Data to be transmitted from the first process to the second process is encrypted by the first process and written to the Java output stream. The data is transmitted to the second network node. Then the data is read from the Java input stream by th second process and decrypted.

FIG. 2




EP 0 887 981 A2

## Description

### FIELD OF THE INVENTION

The invention relates to data security, and more specifically, to a method and apparatus for providing layer-independent security in network communications.

### BACKGROUND OF THE INVENTION

Some communication networks, particularly complex ones, support multiple communication protocols or "layers." Each layer specifies some functionality or " service" of the network and interacts with the layers immediately above and below, using services of the layer immediately below, while providing services to the layer immediately above. The lowest layer in a communication network typically governs direct communication between the hardware at different network nodes, while the highest layer handles direct communication with application programs executing on the network nodes.

The layered approach to implementing communication networks simplifies the creation and modification of complex communication architectures by providing for incremental changes on a layer-by-layer basis, which are transparent to other layers in the architecture. Two examples of layered communication protocols are the Transmission Control Protocol/Internet Protocol (TCP/IP), which has five layers, and the International Standards Organization's (ISO) Open Systems Interconnection (OSI) Reference Model (RM), which has seven layers.

The proliferation of communication networks and increased frequency of security breaches has underscored the importance of providing secure network communications. Many communication networks depend upon a secure communication connection or " channel" to maintain security. In the context of secure communication networks, a secure communication channel is a connection which provides for the encryption, authentication or otherwise secure transmission of data between network nodes.

Sometimes, setup negotiation is used to establish security for a communication channel. In the context of network communications, setup negotiation refers to specifying and agreeing to the details about security for a communication channel, such as the details of a particular encryption scheme to be used. Once setup negotiation is complete, all communication during the session conforms to the agreed upon security protocol, which provides secure communication.

Setup negotiation is an effective tool for providing secure communication during a communication session. However, when the amount of information included in each session is small, for example when a session contains only a single message, then the overhead attributable to setup negotiation can adversely affect communication performance. Moreover, some communication architectures do not include a session layer, which requires that a session layer be added to support session type security, further degrading performance.

Another approach for providing a secure communication channel involves encrypting or encoding data at a specific layer on a transmitting network node and then decrypting or decoding the data at a corresponding layer on a destination network node. Encrypting data at a specific layer typically involves applying an encryption algorithm based upon the format of data at a particular layer. Header data added by higher layers is also encrypted. Layer-specific encryption is particularly useful in datagram-based or packet-based networks which are typically sessionless and encapsulate data in datagram packets or some other type of data packet. For example, header data may be added to a data packet so that the data packet conforms to a particular format. This approach also provides for multiple encryptions to be performed at different layers.

Although layer-specific encryption can provide a secure communication channel while avoiding the overhead penalty associated with setup negotiation, it does have several limitations. First, all encryption and decryption must occur at the same corresponding layer on both the transmitting and receiving network nodes, according to the specific protocol supported by that layer. For example, Simple Key Management for Internet Protocols (SKIP) is designed to be used with intemet protocol packets at the network layer, which requires internet layer specific function calls. On the other hand, Netscape Communications Corporation's Secure Sockets Layer (SSL) is designed to be used at the (Unix) socket layer and requires socket layer-specific function calls to encrypt and decrypt data. The result is that one application implementing security according to SKIP cannot interact with another application implementing security according to SSL.

In addition, layer-specific encryption can be difficult to employ in object-oriented environments because of the inherent level of abstraction required. For example, some layers operate on data bytes, which often is a much lower level than objects in an object oriented environment.

In view of both the need to provide secure communication channels and the limitations in the prior approaches, an approach for providing a secure communication channel which does not rely upon layer-specific encryption and which does not require setup negotiation is highly desirable.

### SUMMARY OF THE INVENTION

According to one aspect of the invention, a method provides communication protocol-indep ndent security for data transmitted betwe n a first process, ex cuting on a first network node, and a second process, ex cuting on a second network node. Both the first network node and the second network node each support at

least one common communication protocol. According to the method, a communication channel is establish d between the first network node and the second network node. Then, a first stream is established between the first process and the communication channel.

In the context of the invention, a " stream" is an abstraction which refers to the transfer or " flow" of data, in any format, from a single source, to a single destination. A stream typically flows through a channel or connection between the sender and receiver, in contrast to data packets, which are typically individually addressed and which may be routed independently to multiple recipients. Hence, an application can write data to, or read data from, a stream without knowing the actual destination or source, respectively, of the data.

After the first stream is established between the first process and the communication channel, a second stream is established between the second process and the communication channel. Data to be transmitted between the first and second processes is encrypted. The encryption of the data is independent of the communication protocol supported by the first network node. The encrypted data is then written to the first stream which causes the encrypted data to be transmitted from the first network node to the second network node. The encrypted data is read from the second stream and then decrypted to obtain decrypted data which is identical to the data on the first network node before the data was encrypted.

## BRIEF DESCRIPTION OF THE DRAWINGS

The invention is illustrated by way of example, and not by way of limitation, in the figures of the accompanying drawings and in which like reference numerals refer to similar elements and in which:

Figure 1 is a block diagram of a multi-layered communication network according to an embodiment of the invention;

Figure 2 is a block diagram of a multi-layered communication network according to another embodiment of the invention;

Figure 3 illustrates a stream format according to an embodiment of the invention;

Figure 4 is a flow chart illustrating a method for providing layer-independent secure communication in a multi-layered communication network according to an embodiment of the invention;

Figure 5 is a block diagram of a Java secure channel arrangement according to an embodiment of the invention; and

Figure 6 is a block diagram of a computer system on which the invention may be implemented.

## DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

A method and apparatus for providing layer-independent secure communications in a multi-layered communication network is described. In the following description, for the purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the invention. However, the invention may be practiced without these specific details. In other instances, well-known structures and devices are illustrated in block diagram form in order to avoid unnecessarily obscuring the invention.

## FUNCTIONAL OVERVIEW

The invention provides a method and apparatus for providing layer-independent secure communications in a multi-layered communication network. In general, a communication channel or connection is first established between a first multi-layered network node and a second multi-layered network node. Then, a first stream is established between a first process, executing on the first multi-layered network node, and the communication channel. A second stream is then established between a second process, executing on the second multi-layered network node and the communication channel. Then, the first process performs a layer-independent encryption of data to be transmitted between the first and second multi-layered network nodes and then writes the encrypted data to the first stream, which causes the encrypted data to be transmitted to the second multi-layered network node. Then, the encrypted data is read by the second process from the second stream and decrypted so that the decrypted data is identical to the data on the first multi-layered network node prior to being encrypted.

Figure 1 illustrates a multi-layered communication network 100 to which the invention is applicable. In general, multi-layered communication network 100 includes multi-layered nodes 102, 104, communicatively coupled by transmission medium 106. Although multi-layered communication network 100 may resemble the International Standards Organization (ISO) Open Systems Interconnection (OSI) Reference Model (RM), the invention is applicable to any multi-layered communication network.

A process 108 executes on multi-layered node 102 while a process 110 executes on multi-layered node 104. Multi-layered node 102 supports a multi-layered communication hierarchy 112, where each identified layer supports a particular communication protocol. Each layer in hierarchy 112 offers certain services to the higher layers while shielding the higher layers from the details of how those services are actually implemented. Multi-layered nod 104 also supports a multi-layered communication hierarchy 114, which includes layer corresponding to the layers in hierarchy 112. All data trans-

mitted from process 108 to transmission medium 106 conforms to all communication protocols supported by hierarchy 112.

For example, to transmit data 116 from process 108 to transmission medium 106, data 116 must first conform to an application protocol specified by application layer 118 on multi-layered node 102. According to one embodiment of the invention, this requires that data 116 be formatted according to application layer 118 protocol and that an application protocol header AH be appended to the front end of data 116 which specifies the format of data 116.

This process is repeated for each layer in hierarchy 112. According to one embodiment of the invention, the formatting of data 116 according to a data link layer 120 involves the addition of both a header portion DH and a trailer portion DT to a data portion 122. It should be noted that data link layer 120 is not aware of which portion of data portion 122 corresponds to data 116 and which portion represents formatting information of higher layers. Data link layer 120 formats the entire data portion 122 without regard to which portion may be "real" data 116 and which portion is formatting information added by higher layers in hierarchy 112.

When messages are received by multi-layered node 104 from transmission medium 106, a reverse process occurs. Since messages must conform to application layer protocol before being processed by process 110, any formatting information attributable to layers below application layer 128 must be removed.

As previously discussed, one approach for providing secure communication between process 108 and process 110 is to have processes 108, 110 perform setup negotiation prior to transmitting data. However, this approach can adversely affect data throughput, particularly when the setup negotiation is performed on a packet-by-packet basis.

Another previously discussed approach is to encrypt the data at one of the layers in hierarchy 112 on multi-layered node 102 before transmitting the data on transmission medium 106. Then, after the encrypted data is received on node 104, the data is decrypted at the corresponding layer in hierarchy 114 on multi-layered node 104 before the data is received by process 110. For example, data may be encrypted at the network layer 124 on multi-layered node 102 and then decrypted at network layer 126 on multi-layered node 104 on a packet-by-packet basis. Although this approach is robust from a security standpoint, the data must be decrypted at the same layer at which the data was encrypted.

LAYER-INDEPENDENT SECURITY

An approach which provides layer-independent secure network communication in a multi-layered communication network according to an embodiment of the invention is illustrated by the block diagram of Figure 2. A multi-layered communication network 200 includes multi-layered nodes 202, 204 which are communicatively coupled by a transmission medium 206. A process 208 executes on multi-layered node 202 while a process 210 executes on multi-layered node 204.

Multi-layered nodes 202, 204 each support one or more communication layers (protocols) including socket layers 212, 214, respectively. Socket layers 212, 214 provide an interface between processes 208, 210, respectively, and transmission medium 206. Multi-layered nodes 202, 204 may support addition layers (not illustrated) both above and below socket layers 212, 214. Accordingly, socket layers 212, 214 each include sockets (not illustrated), which are end points similar to an OSI Transport Service Access Point (TSAP), and which provide a connection between layers above and below socket layers 212, 214. In addition, a Java secure channel 216 is provided between node 202 and node 204. Java security channel 216 provides for the layer-independent encryption of high level data constructs such as objects.

Generally, according to an embodiment of the invention, layer-independent security for communications between process 208 and process 210 is provided by process 208 encrypting data which is then written to a Java output stream 218. A Java stream is a stream which provides for the transfer of low level data constructs, such as bytes, as well as high level data constructs, such as serialized objects, between a source and a destination. The data is then conformed to a socket layer protocol by socket layer 212 and written to transmission medium 206. The data is then processed according to socket layer protocol by socket layer 214 and read from a Java input stream 220 by process 210 and finally decrypted by process 210.

Encryption of stream data according to embodiments of the invention is by definition layer-independent and provides a level of abstractness which is compatible with many abstract processes and languages which support streams, such as object oriented languages. Besides the layer-independent data encryption performed by process 208, additional (layer-dependent) encryption may be provided at any layer in node 202, with decryption being performed at the corresponding peer layer in node 204.

The data format of object output stream 218 and object input stream 220 is illustrated in Figure 3. Generally, stream format 300 is an abstract message format which is self-contained and layer-independent. Stream format 300 includes 1 to N variable length messages (M1, M2... Mn). Each message (M1, M2... Mn) includes a header portion (H1, H2...Hn) and a data portion (DATA1, DATA2...DATAn). According to one embodiment of the invention, each header portion (H1, H2... Hn) specifies the length of the associated data portion (D1, D2...Dn) and also includes encryption key/authentication information which eliminates the need for s tup negotiation. However, certain encryption key/auth ntication information is established once during system

setup so that recipients of the messages (M1, M2...Mn) can decrypt data contained in the data portion (D1, D2... Dn) of each message (M1, M2...Mn).

The flexibility of stream format 300 of the invention provides for the implementation of various encryption/authentication approaches and is not limited to the particular encryption/authentication approach described herein. In addition, since stream format 300 is layer independent, various data formats may be employed without departing from the scope of the invention.

The specific steps for providing layer-independent secunty of network communication according to an embodiment of the invention are now described with reference to both the block diagram of Figure 2 and the flow chart of Figure 4. Generally, the steps are described in the context of an object oriented programming method associated with an object, contained in process 208, which invokes a method associated with a remotely located object contained in process 210. In the non-object oriented context, this is very similar to process 208 issuing a remote procedure call (RPC) to invoke a process remotely located on multi-layered node 204. For purposes of explanation, the data transmitted by the method associated with the object contained in process 208 which invokes the method associated with the remotely located object contained in process 210 is hereinafter referred to as the " object data."

After starting in step 400, in step 402, multi-layered nodes 202, 204 establish an encryption/authentication approach during system setup. Unlike traditional setup negotiation which must be continuously re-negotiated, such as on a per session basis, the agreed upon encryption/authentication approach established between multi-layered nodes 202, 204 only needs to be set up once during system setup, or when either multi-layered node 202, 204 is connected to another node and the security techniques described herein are to be employed with that other node.

In step 404, a Java secure channel 216 is established between node 202 and node 204. According to one embodiment of the invention, Java secure channel 216 is an object class which is defined and invoked by process 208.

In step 406, object output stream 218 is established between process 208 and socket layer 212, and in step 408, object input stream 220 is established between socket layer 214 and process 210. According to one embodiment of the invention, object output stream 218 is an object class defined by process 208 while object input stream 220 is an object class defined by process 210.

In step 410, the object data to be transmitted from process 208 to process 210 is serialized, sometimes referred to as " flattening the object," and then encrypt d in step 412 based upon the encryption/authentication approach established in step 402.

In step 414, the object data (serialized and encrypted) is written to object output stream 218, which is received by socket layer 212 and formatted according to socket layer protocol. In step 416, the object data is transmitted from socket layer 212 of multilayered node 202 to socket layer 214 of multi-layered node 204 over transmission medium 206.

As previously discussed, multi-layered node 202 is illustrated as having a single layer, socket layer 212, while multi-layered node 204 is illustrated as having a single layer, socket layer 214, for purposes of explanation. However, multi-layered nodes 202, 204 may be multi-layered and contain other layers above and below socket layers 212, 214. Consequently, although according to an embodiment of the invention, the object data is transmitted onto transmission medium 206 in the format illustrated in Figure 3, it is understood that additional formatting of the object data may be performed according to various other communication protocols contained in multi-layered nodes 202, 204. For example, if multi-layered node 202 also supports Internet protocol (IP), then each message (M1, M2...Mn) illustrated in Figure 3 would also contain IP header information.

After the object data is received by socket layer 214, the object data is read from object input stream 220 by process 210 in step 418. In step 420, the object data is decrypted according to the encryption/authentication approach established in step 402. Then, in step 422, the object data is de-serialized and the method associated with the object remotely located in process 210 is executed. Finally, the process is complete in step 424.

Although embodiments of the invention have been described in the context of encrypting and decrypting object data by processes 208, 210, which are effectively above all of the layers supported by multi-layered nodes 202, 204, respectively, data may be encrypted and decrypted at any layer supported by multi-layered nodes 202, 204, since the encryption of data is performed before the data is written to a stream and is therefore layer-independent.

For example, referring again to Figure 1, according to another embodiment of the invention, process 108 encrypts data 116 and then writes data 116 to a stream (not illustrated) which is formatted according to the protocol hierarchy 112 and transmitted to multi-layered node 104 on transmission medium 106. Since data 116 was encrypted at the stream level, data 116 may be decrypted at any layer in hierarchy 114, so long as data 116 can be extracted from the data stream. Typically, the size and position of data 116 within a data chunk is known which allows data 116 to be extracted from a data chunk even though the data chunk contains protocol specific information from higher layers. However, if data 116 is encrypted at any other layer in hierarchy 112, then data 116 must first be decrypted at a corresponding lay- r in hierarchy 114.

According to another embodiment of the invention, a stream is connected to several other protocol-specific streams to support the broadcasting or multi-casting of encrypted information. Figure 5 illustrates an arrange-

ment 500 which includes a stream 502 according to an embodiment of the invention, connected via connectors 504, to intelligent converters 506, which convert stream 502 into protocol-specific streams 508 such as file I/O, object I/O, and socket I/O streams. Converters 506 have the capability to extract the data portion from stream 502 to support streams 508 at any protocol layer.

According to arrangement 500, any number of protocol-specific streams 508 may be connected to stream 502. In addition, the headers of messages in stream 502 may contain destination-specific encryption/authentication information. For example, stream 502 may contain an encryption/authentication value A, while a recipient of one of the protocol-specific streams 508 holds a key value X, making the decryption of stream 502 a function of A and X (key=f(A,X)). Likewise, similar keys may be developed for the other protocol-specific streams 508.

## HARDWARE OVERVIEW

Figure 6 is a block diagram which illustrates a computer system 600 upon which an embodiment of the invention may be implemented. Computer system 600 includes a bus 602 or other communication mechanism for communicating information, and a processor 604 coupled with bus 602 for processing information. Computer system 600 also includes a main memory 606, such as a random access memory (RAM) or other dynamic storage device, coupled to bus 602 for storing information and instructions to be executed by processor 604. Main memory 606 also may be used for storing temporary variables or other intermediate information during execution of instructions by processor 604. Computer system 600 also includes a read only memory (ROM) 608 or other static storage device coupled to bus 602 for storing static information and instructions for processor 604. A storage device 610, such as a magnetic disk or optical disk, is also provide and coupled to bus 602 for storing information and instructions.

Computer system 600 may also be coupled via bus 602 to a display 612, such as a cathode ray tube (CRT), for displaying information to a computer user. An input device 614, including alphanumeric and other keys, is also provided and coupled to bus 602 for communicating information and command selections to processor 604. Another type of user input device is cursor control 616, such as a mouse, a trackball, or cursor direction keys for communicating direction information and command selections to processor 604 and for controlling cursor movement on display 612. This input device typically has two degrees of freedom in two axes, a first axis (e.g., x) and a second axis (e.g., y), which allows the device to specify positions in a plane.

The invention is related to the use of computer system 600 to provide layer-independent secure network communication. According to one embodiment of the invention, layer-independent secure network communication is provided by computer system 600 in response to processor 604 executing sequences of instructions contained in main memory 606. Such instructions may be read into main memory 606 from another computer-readable medium, such as storage device 610. However, the computer-readable medium is not limited to devices such as storage device 610. For example, the computer-readable medium may include a floppy disk, a flexible disk, hard disk, magnetic tape, or any other magnetic medium, a CD-ROM, any other optical medium, a RAM, a PROM, and EPROM, a FLASH-EPROM, any other memory chip or cartridge, or any other medium from which a computer can read. Execution of the sequences of instructions contained in main memory 606 causes processor 604 to perform the process steps previously described. In alternative embodiments, hard-wired circuitry may be used in place of or in combination with software instructions to implement the invention. Thus, embodiments of the invention are not limited to any specific combination of hardware circuitry and software.

Computer 600 also includes a communication interface 618 coupled to bus 602. Communication interface 608 provides a two-way data communication coupling to a network link 620 to a local network 622. For example, if communication interface 618 is an integrated services digital network (ISDN) card or a modem, communication interface 618 provides a data communication connection to the corresponding type of telephone line. If communication interface 618 is a local area network (LAN) card, communication interface 618 provides a data communication connection to a compatible LAN. Wireless links are also possible. In any such implementation, communication interface 618 sends and receives electrical, electromagnetic or optical signals which carry digital data streams representing various types of information.

Network link 620 typically provides data communication through one or more networks to other data devices. For example, network link 620 may provide a connection through local network 622 to a host computer 624 or to data equipment operated by an Internet Service Provider (ISP) 626. ISP 626 in turn provides data communication services through the world wide packet data communication network now commonly referred to as the "Internet" 628. Local network 622 and Internet 628 both use electrical, electromagnetic or optical signals which carry digital data streams. The signals through the various networks and the signals on network link 620 and through communication interface 618, which carry the digital data to and from computer 600 are exemplary forms of carrier waves transporting the information.

Computer 600 can send messages and receive data, including program code, through the network(s), network link 620 and communication interface 618. In the Internet example, a server 630 might transmit a requested code for an application program through Internet 628, ISP 626, local network 622 and communication inter-

face 618. In accord with the invention, one such downloaded application provides for the synchronization of threads using selective object locking as described herein.

The received code may be executed by processor 604 as it is received, and/or stored in storage device 610, or other non-volatile storage for later execution. In this manner, computer 600 may obtain application code in the form of a carrier wave.

Although the invention has been described in the context of connection-based communication architectures, the invention is also applicable to sessionless datagram or packet based communication architectures.

The invention provides several advantages over prior approaches for implementing secure network communications. Most importantly, security is implemented using streams which are layer independent. This allows an encrypted stream to be decrypted at any layer without requiring the use of layer specific calls to perform the decryption, which provides greater flexibility than prior approaches. For example, an encrypted stream transmitted by a sending node may be decrypted by a firewall connection at the network (packet) layer having knowledge of the encryption approach negotiated during system setup. Moreover, this approach does not affect existing encryption being carried out at various layers. The approach of the invention avoids the setup negotiation which can significantly degrade communication performance in certain situations.

In the foregoing specification, the invention has been described with reference to specific embodiments thereof. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention. The specification and drawings are, accordingly, to be regarded in an illustrative rather than a restrictive sense.

## Claims

1. A method for providing communication protocol-independent security for data transmitted between a first process, executing on a first network node, and a second process, executing on a second network node, wherein the first network node and the second network node each support at least one common communication protocol, the method comprising the steps of:

   a) establishing a communication channel between the first network node and the second network node;
   b) establishing a first stream between the first process and the communication channel;
   c) establishing a s cond stream between the second process and the communication channel;
   d) encrypting data to be transmitted between the first and second processes, the encrypting of the data being independent of the at least one communication protocol supported by the first network node;
   e) writing the encrypted data to the first stream;
   f) causing the encrypted data to be transmitted from the first network node to the second network node;
   g) reading the encrypted data from the second stream; and
   h) decrypting the encrypted data to obtain decrypted data which is identical to the data on the first network node before the data was encrypted.

2. The method of Claim 1, further including the steps of

   a) performing a communication protocol-specific encryption of the data on the first network node, and
   b) performing a communication protocol-specific decryption of the data on the second network node.

3. The method of Claim 1, wherein the communication channel is a Java secure channel,

   wherein the first stream is a first Java stream,
   wherein the second stream is a second Java stream,
   wherein the step of establishing a communication channel between the first and second network nodes further comprises the step of establishing a Java secure channel between the first and second network nodes,
   wherein the step of establishing a first stream between the first process and the communication channel further comprises the step of establishing a first Java stream between the first process and the Java secure channel,
   wherein the step of establishing a second stream between the second process and the communication channel further comprises the step of establishing a second Java stream between the second process and the Java secure channel,
   wherein the step of writing the encrypted data to the first stream further comprises the step of writing the encrypted data to the first Java stream, and
   wherein the step of reading the encrypted data from the second stream further comprises the step of reading the encrypted data from the second Java str am.

4. The method of Claim 1, wh rein the communication channel is a Java secure channel, wherein the first

stream is a Java stream,

wherein the second stream is a Java stream, wherein the method further comprises the step of connecting the Java secure channel to a third Java stream, and
wherein the third Java stream provides for the transmission of data according to a specific communication protocol.

5. A computer-readable medium having stored thereon a plurality of sequences of instructions for providing communication protocol-independent security for data transmitted between a first process, executing on a first network node, and a second process, executing on a second network node, wherein the first network node and the second network node each support at least one common communication protocol, the plurality of sequences of instructions including sequences of instructions which, when executed by one or more processors, cause the one or more processors to perform the steps of:

a) establishing a communication channel between the first network node and the second network node;
b) establishing a first stream between the first process and the communication channel;
c) establishing a second stream between the second process and the communication channel;
d) encrypting data to be transmitted between the first and second processes, the encrypting of the data being independent of the communication protocols supported by the first network node;
e) writing the encrypted data to the first stream;
f) causing the encrypted data to be transmitted from the first network node to the second network node;
g) reading the encrypted data from the second stream; and
h) decrypting the encrypted data to obtain decrypted data which is identical to the data on the first network node before the data was encrypted.

6. The computer-readable medium of Claim 5, wherein the computer-readable medium further includes instructions for performing the steps of

a) performing a communication protocol-specific encryption of the data on the first network node, and
b) performing a communication protocol-specific decryption of the data on the second network node.

7. The computer-readable medium of Claim 5, wherein the first stream is a first Java stream,

wherein the second stream is a second Java stream,
wherein the step of establishing a communication channel between the first and second network nodes further comprises the step of establishing a Java secure channel between the first and second network nodes,
wherein the step of establishing a first stream between the first process and the communication channel further comprises the step of establishing a first Java stream between the first process and the Java secure channel,
wherein the step of establishing a second stream between the second process and the communication channel further comprises the step of establishing a second Java stream between the second process and the Java secure channel,
wherein the step of writing the encrypted data to the first stream further comprises the step of writing the encrypted data to the first Java stream, and
wherein the step of reading the encrypted data from the second stream further comprises the step of reading the encrypted data from the second Java stream.

8. The computer-readable medium of Claim 5, wherein the communication channel is a Java secure channel,

wherein the first stream is a Java stream,
wherein the second stream is a Java stream,
wherein the computer-readable medium further includes instructions for connecting the Java secure channel to a third Java stream, and
wherein the third Java stream provides for the transmission of data according to a specific communication protocol.

9. A communication network providing communication protocol-independent secure communication between a first network node and a second network node, wherein the first network node and the second network node each support at least one common communication protocol, wherein the first network node is communicatively coupled to the second network node by a communication channel, the communication network comprising:

a) a first process executing on the first network node, wherein the first process provides for the communication protocol-independent encryption of data;
b) a first stream which provides for the transfer

of encrypted data between the first process and the communication channel;
c) a second process executing on the second network node; and
d) a second stream which provides for the transfer of encrypted data between the communication channel and the second process, wherein the second process also provides for the decryption of data which has been encrypted by the first process.

10. The communication network of Claim 9, wherein the second process further includes the capability to decrypt data based upon any communication protocol supported by the second network node.

11. The communication network of Claim 9, wherein the communication channel is a Java secure channel, the first stream is a Java stream and the second stream is a Java stream.

12. The communication network of Claim 11, further comprising a third Java stream connected to the Java secure channel, the third Java stream providing for the transmission of data according to a specific communication protocol.

13. A computer data signal embodied in a carrier wave and representing sequences of instruction which, when executed by one or more processors, provide communication protocol-independent security for data transmitted between a first process, executing on a first network node, and a second process, executing on a second network node, wherein the first network node and the second network node each support at least one common communication protocol by performing the steps of:

a) establishing a communication channel between the first network node and the second network node;
b) establishing a first stream between the first process and the communication channel;
c) establishing a second stream between the second process and the communication channel;
d) encrypting data to be transmitted between the first and second processes, the encrypting of the data being independent of the communication protocols supported by the first network node;
e) writing the encrypted data to the first stream;
f) causing the encrypted data to be transmitted from the first network node to the second network node;
g) reading the encrypted data from the second stream; and
h) decrypting the encrypted data to obtain decrypted data which is identical to the data on the first network node before the data was encrypted.

14. The computer data signal of Claim 13, wherein the computer sequence of instructions further includes instructions for performing the steps of

a) performing a communication protocol-specific encryption of the data on the first network node, and
b) performing a communication protocol-specific decryption of the data on the second network node.

15. The computer data signal of Claim 13, wherein the first stream is a first Java stream,

wherein the second stream is a second Java stream,
wherein the step of establishing a communication channel between the first and second network nodes further comprises the step of establishing a Java secure channel between the first and second network nodes,
wherein the step of establishing a first stream between the first process and the communication channel further comprises the step of establishing a first Java stream between the first process and the Java secure channel,
wherein the step of establishing a second stream between the second process and the communication channel further comprises the step of establishing a second Java stream between the second process and the Java secure channel,
wherein the step of writing the encrypted data to the first stream further comprises the step of writing the encrypted data to the first Java stream, and
wherein the step of reading the encrypted data from the second stream further comprises the step of reading the encrypted data from the second Java stream.

16. The computer data signal of Claim 13, wherein the communication channel is a Java secure channel,

wherein the first stream is a Java stream,
wherein the second stream is a Java stream,
wherein the computer sequence of instructions further includes instructions for connecting the Java secure channel to a third Java stream, and
wherein the third Java stream provides for th transmission of data according to a specific communication protocol.

17. A method for providing communication protocol-in-

dependent security for data transmitted by a process executing on a network node, the method comprising the steps of:

a) establishing a stream between the process and a communication channel;
b) encrypting data to be transmitted by the process, the encrypting of the data being independent of a communication protocol supported by the network node;
c) writing the encrypted data to the stream; and
d) causing the encrypted data to be transmitted from the network node to the communication channel.

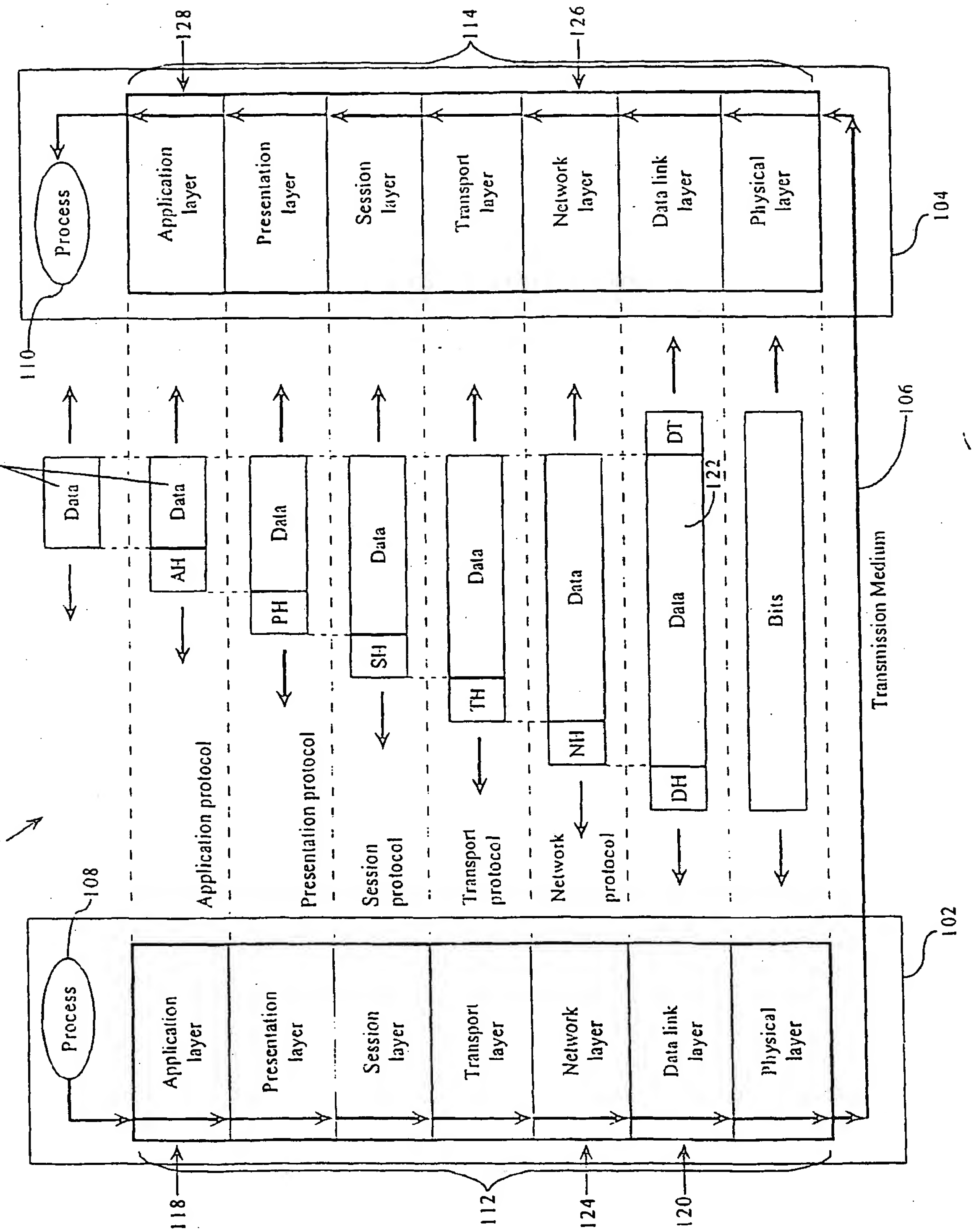
**18.** The method of Claim 17, wherein the communication channel is a Java secure channel,

wherein the stream is a first Java stream,
wherein the step of establishing a stream between the process and the communication channel further comprises the step of establishing a Java stream between the process and the Java secure channel, and
wherein the step of writing the encrypted data to the stream further comprises the step of writing the encrypted data to the Java stream.

**19.** The method of Claim 17, wherein the communication channel is a Java secure channel, wherein the stream is a Java stream,

wherein the method further comprises the step of connecting the Java secure channel to a second Java stream, and
wherein the second Java stream provides for the transmission of data according to a specific communication protocol.

FIG. 1


Process
Application layer
Presentation layer
Session layer
Transport layer
Network layer
Data link layer
Physical layer
Process
Application layer
Presentation layer
Session layer
Transport layer
Network layer
Data link layer
Physical layer
Data
AH
Data
PH
Data
SH
Data
TH
Data
NH
Data
DH
Data
DT
Bits
Application protocol
Presentation protocol
Session protocol
Transport protocol
Network protocol
Transmission Medium
100
102
104
106
108
110
112
114
116
118
120
122
124
126
128

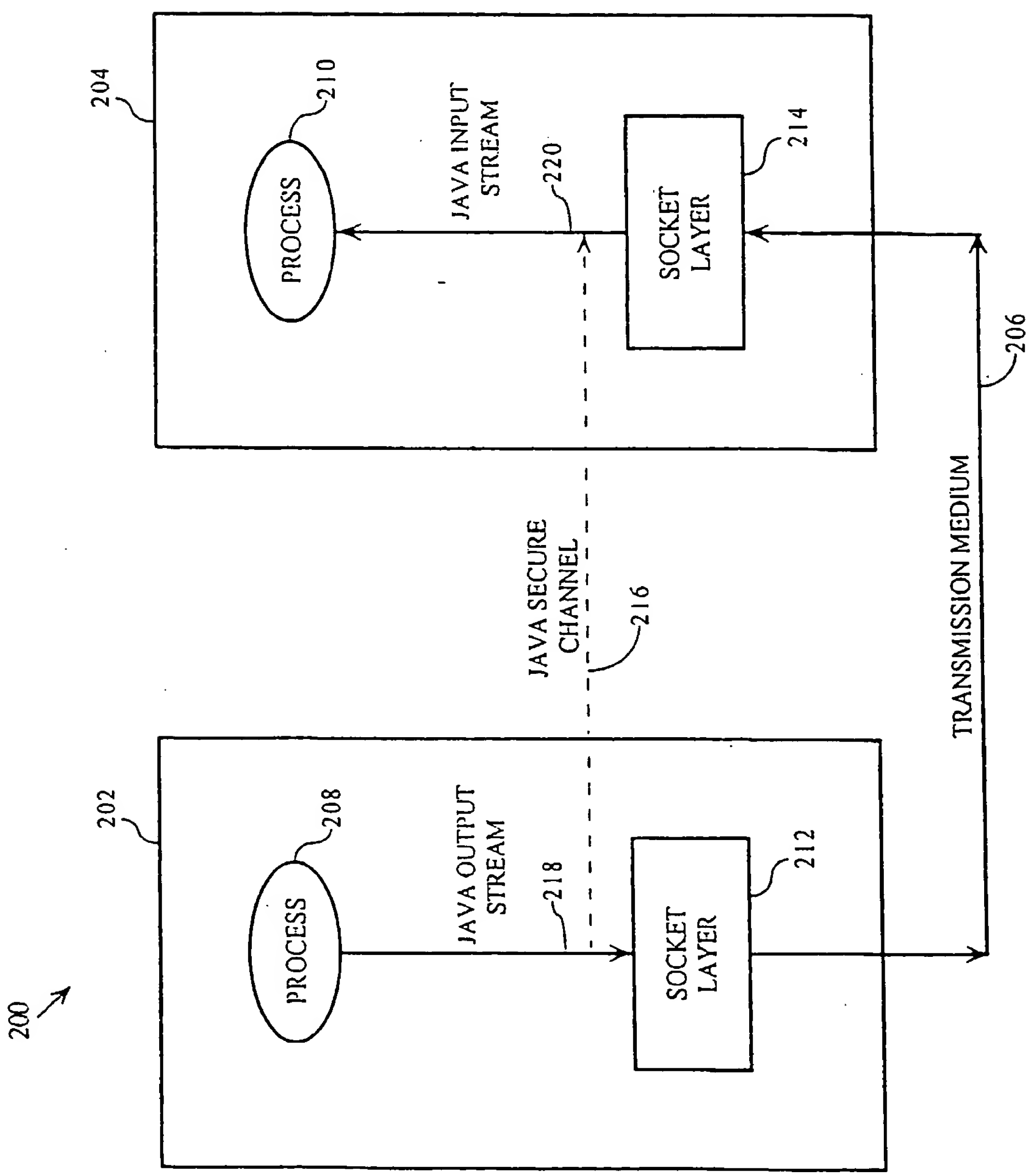

200
202
PROCESS
208
JAVA OUTPUT STREAM
218
SOCKET LAYER
212
JAVA SECURE CHANNEL
216
TRANSMISSION MEDIUM
206
204
PROCESS
210
JAVA INPUT STREAM
220
SOCKET LAYER
214

FIG. 2

FIG. 3

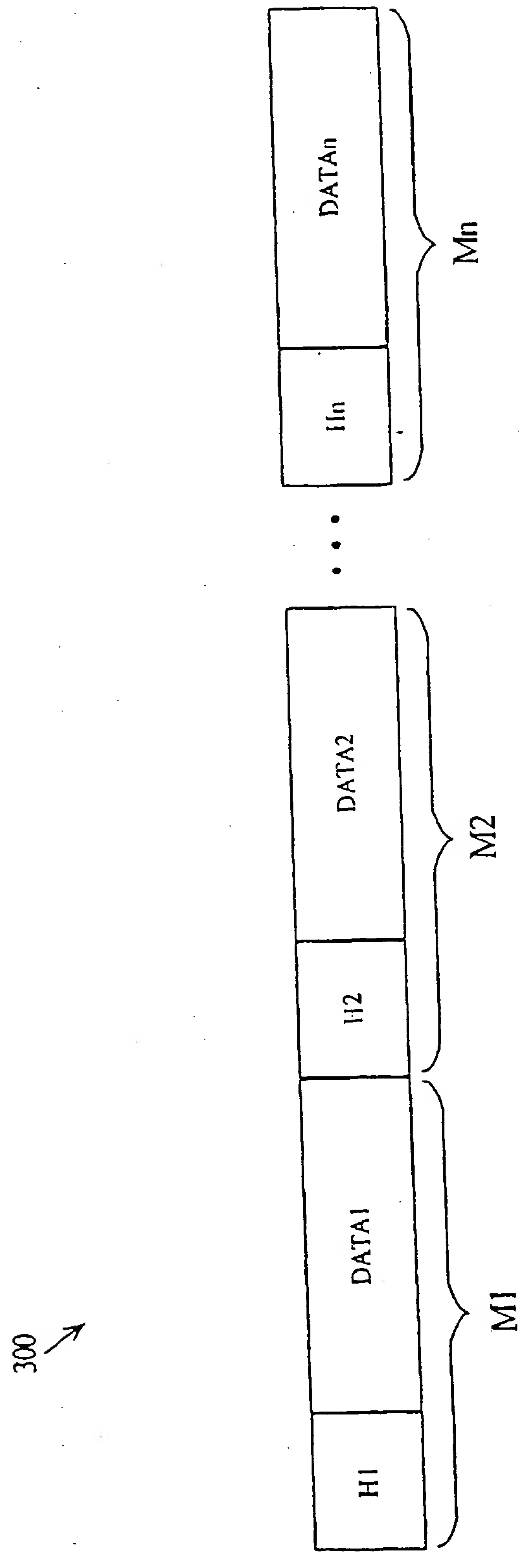


FIG. 4

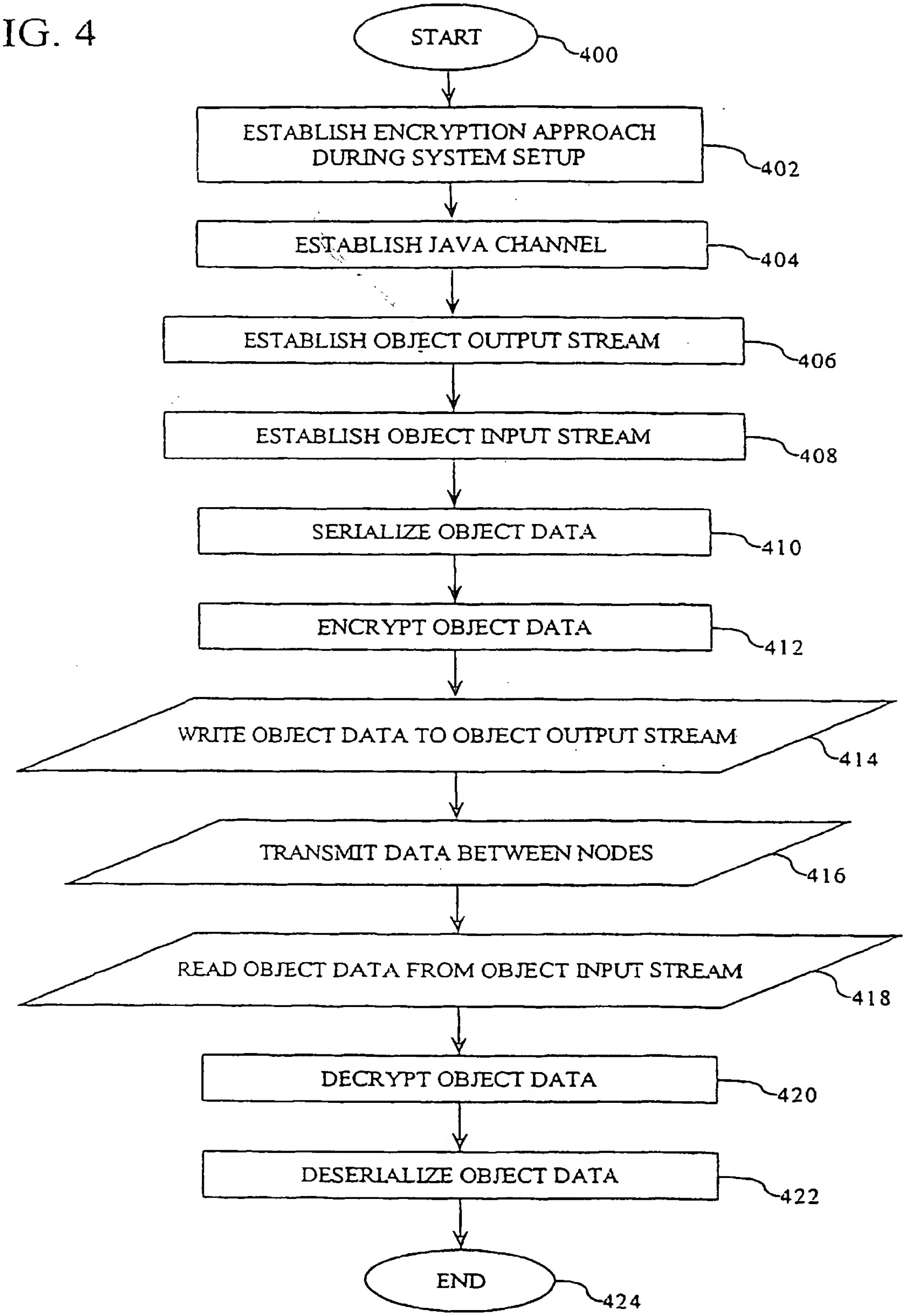
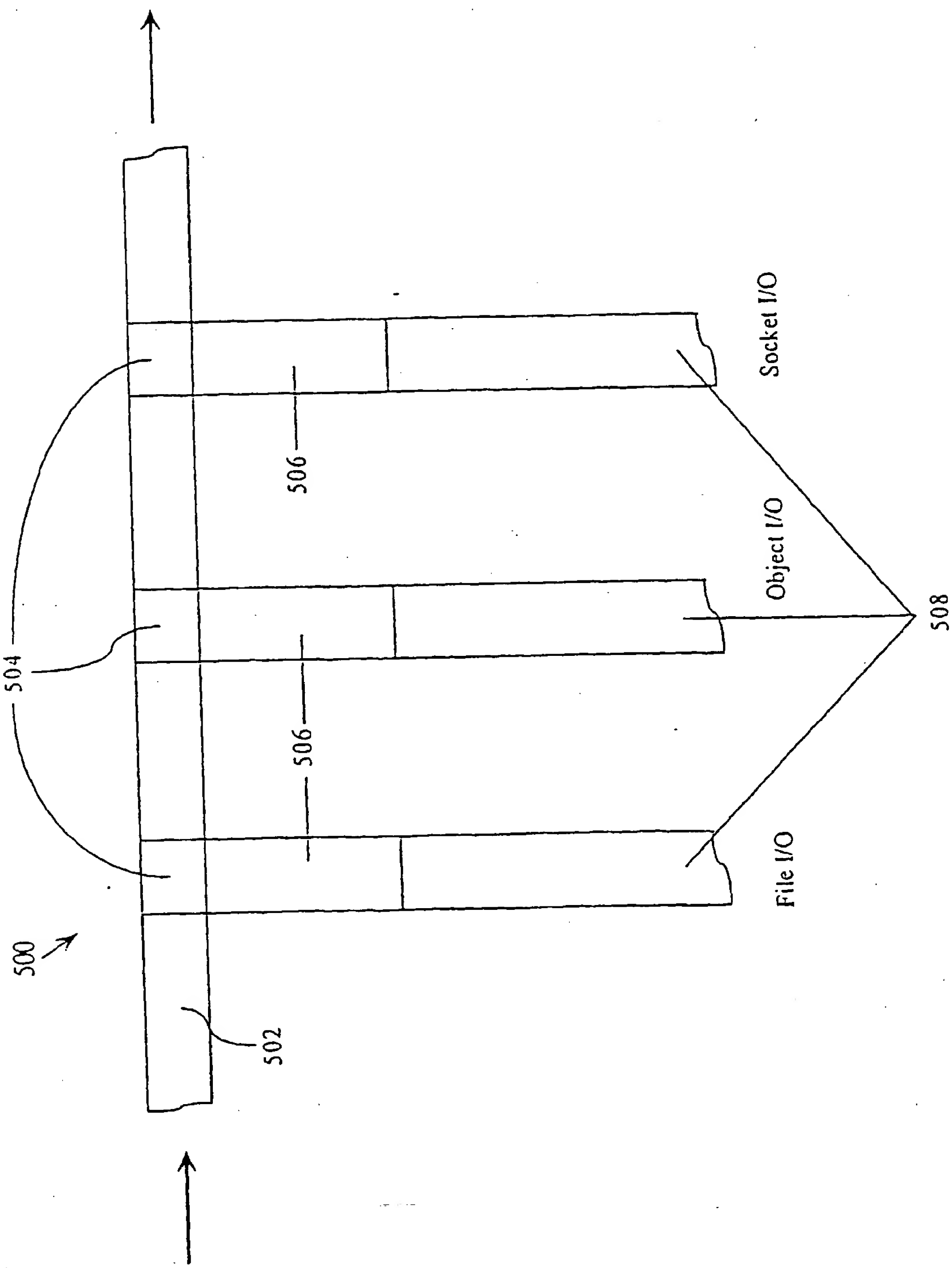

START
400
ESTABLISH ENCRYPTION APPROACH DURING SYSTEM SETUP
402
ESTABLISH JAVA CHANNEL
404
ESTABLISH OBJECT OUTPUT STREAM
406
ESTABLISH OBJECT INPUT STREAM
408
SERIALIZE OBJECT DATA
410
ENCRYPT OBJECT DATA
412
WRITE OBJECT DATA TO OBJECT OUTPUT STREAM
414
TRANSMIT DATA BETWEEN NODES
416
READ OBJECT DATA FROM OBJECT INPUT STREAM
418
DECRYPT OBJECT DATA
420
DESERIALIZE OBJECT DATA
422
END
424

FIG. 5



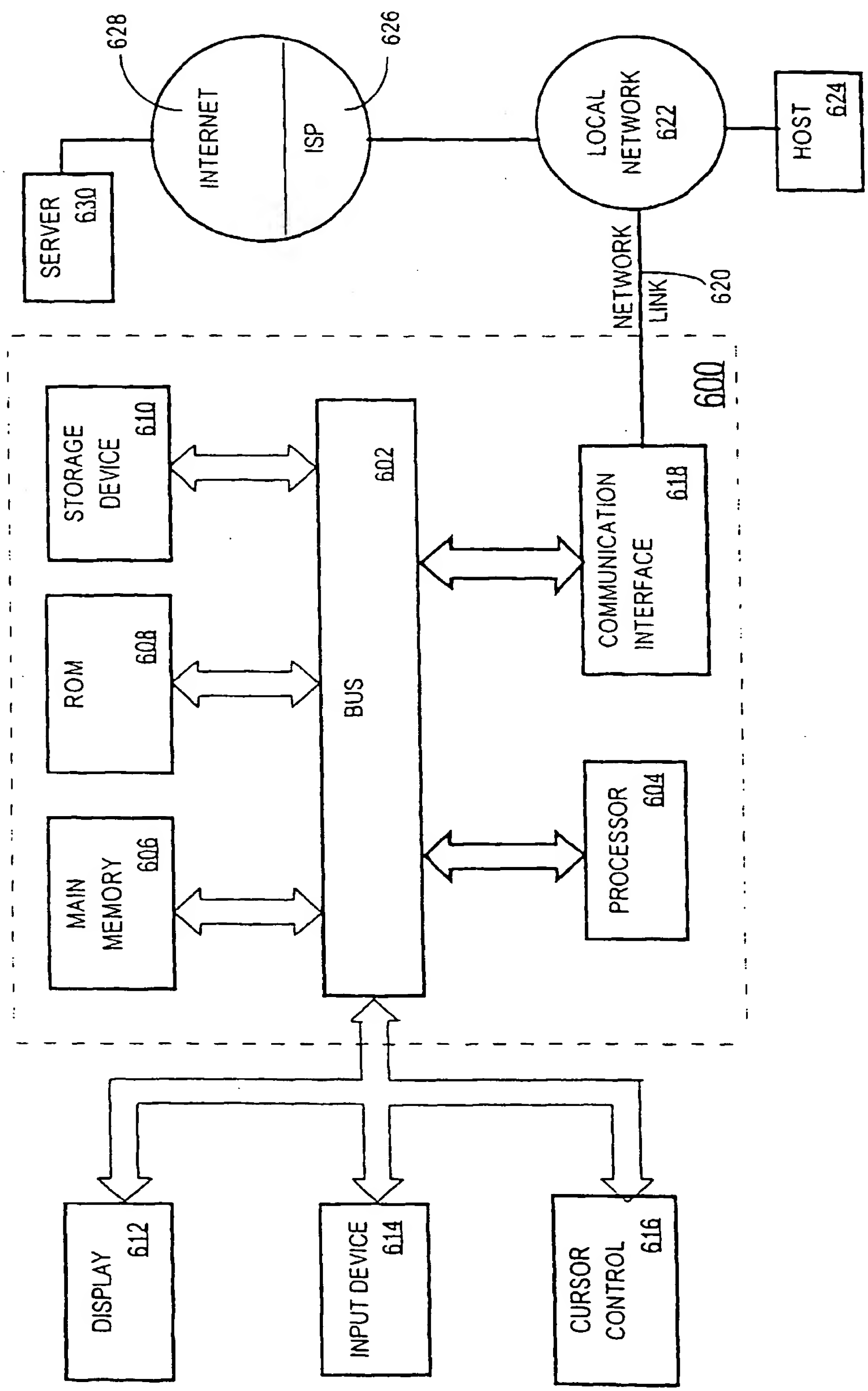

SERVER
630
INTERNET
628
ISP
626
LOCAL
NETWORK
622
HOST
624
NETWORK
LINK
620
600
STORAGE
DEVICE
610
ROM
608
MAIN
MEMORY
606
BUS
602
COMMUNICATION
INTERFACE
618
PROCESSOR
604
DISPLAY
612
INPUT DEVICE
614
CURSOR
CONTROL
616

FIG. 6

(19) Europäisch s Patentamt
European Pat nt Offi e
Offic urop´ n d s br vets

(11) EP 0 887 981 A3

(12) EUROPEAN PATENT APPLICATION

(88) Date of publication A3:
28.07.1999 Bulletin 1999/30

(43) Date of publication A2:
30.12.1998 Bulletin 1998/53

(21) Application number: 98304869.5

(22) Date of filing: 19.06.1998

(51) Int Cl.6: H04L 29/06

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

(30) Priority: 26.06.1997 US 883636

(71) Applicant: SUN MICROSYSTEMS, INC.
Mountain View, California 94043-1100 (US)

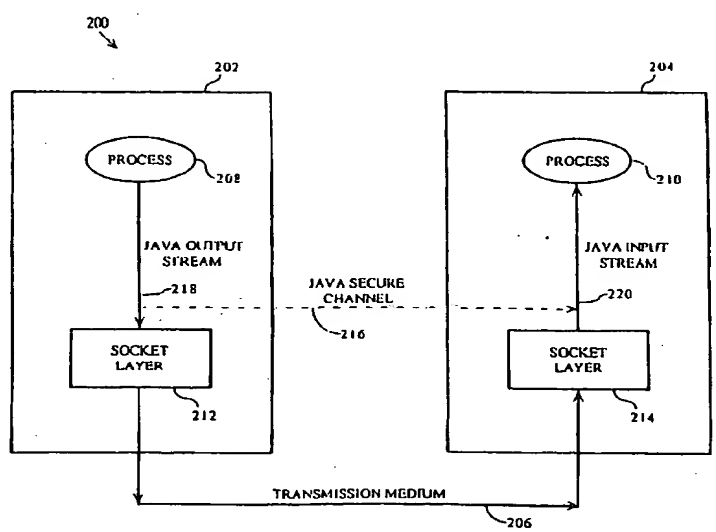
(72) Inventor: Gong, Li
Menlo Park, California 94025 (US)

(74) Representative: Read, Matthew Charles et al
Venner Shipley & Co.
20 Little Britain
London EC1A 7DH (GB)

(54) Layer-independent security for communication channels

(57) A method and apparatus for providing layer-independent secure network communication is provided. According to an embodiment of the invention, a transmission medium (206) is provided between a first network node (200) and a second network node (204). Both the first network node and the second network node support at least one common communication protocol. A Java output stream (218) is established between a first process (208) executing on the first network node and the transmission medium. Also, a Java input stream (220) is established between a second process (210) executing on the second multilayered node and the transmission medium. Data to be transmitted from the first process to the second process is encrypted by the first process and written to the Java output stream. The data is transmitted to the second network node. Then the data is read from the Java input stream by the second process and decrypted.

FIG. 2




EP 0 887 981 A3

Printed by Jouve, 75001 PARIS (FR)

BNSDOCID: <EP___0887981A3_I_>

European Patent Office

# EUROPEAN SEARCH REPORT

Application Number
EP 98 30 4869

## DOCUMENTS CONSIDERED TO BE RELEVANT

| Category | Citation of document with indication, where appropriate, of relevant passages | Relevant to claim | CLASSIFICATION OF THE APPLICATION (Int.Cl.6) |
|---|---|---|---|
| X | EP 0 778 512 A (SUN MICROSYSTEMS INC) 11 June 1997<br>* abstract *<br>* page 2, column 2, line 15 - line 44 *<br>* page 3, column 4, line 17 - page 4, column 6, line 39 *<br>* page 6, column 10, line 24 - line 39 * | 1,3-9, 11,13, 15,17,18 | H04L29/06 |
| Y | | 2,10,12, 14,19 | |
| Y | WO 95 25396 A (TRANSACTION TECHNOLOGY INC) 21 September 1995<br>* abstract *<br>* page 13, line 20 - page 18, line 10 *<br>* page 35, line 16 - page 36, line 6 *<br>* page 44, line 12 - page 45, line 25 *<br>* page 47, line 22 - page 48, line 10 *<br>* page 58, line 3 - line 21 *<br>* figure 16 * | 2,10,12, 14,19 | |
| A | EP 0 776 112 A (SUN MICROSYSTEMS INC) 28 May 1997<br>* abstract *<br>* page 2, column 2, line 5 - line 53 *<br>* page 3, column 3, line 45 - column 4, line 8 *<br>* page 4, column 6, line 41 - page 5, column 7, line 3 *<br>* claims 1,2 * | 1,5,9, 13,17 | TECHNICAL FIELDS SEARCHED (Int.Cl.6)<br>H04L |
| A | ZUOUETE A ET AL: "TRANSPARENT AUTENTICATION AND CONFIDENTIALITY FOR STREAM SOCKETS"<br>IEEE MICRO,<br>vol. 16, no. 3, 1 June 1996, pages 34-41,<br>XP000594076<br>* the whole document * | 1,5,9, 13,17 | |

The present search report has been drawn up for all claims

| Place of search | Date of completion of the search | Examiner |
|---|---|---|
| THE HAGUE | 21 May 1999 | Poggio, F |

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

## ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 98 30 4869

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report.
The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

21-05-1999

| Patent document cited in search report | Publication date | Patent family member(s) | Publication date |
|---|---|---|---|
| EP 0778512 A | 11-06-1997 | US 5708709 A | 13-01-1998 |
| | | JP 9288575 A | 04-11-1997 |
| WO 9525396 A | 21-09-1995 | US 5572572 A | 05-11-1996 |
| | | AU 2123495 A | 03-10-1995 |
| | | CA 2185662 A | 21-09-1995 |
| | | CN 1146262 A | 26-03-1997 |
| | | EP 0750817 A | 02-01-1997 |
| EP 0776112 A | 28-05-1997 | US 5680461 A | 21-10-1997 |
| | | AU 6793096 A | 01-05-1997 |
| | | CA 2187970 A | 27-04-1997 |
| | | JP 9270788 A | 14-10-1997 |
| | | US 5850449 A | 15-12-1998 |

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82